

# Projekt

Temat: Aplikacja do obsługi treningów

Wykonali: Yuliia Prysiazhna, Darya Benediktovich, Sebastian Gawron

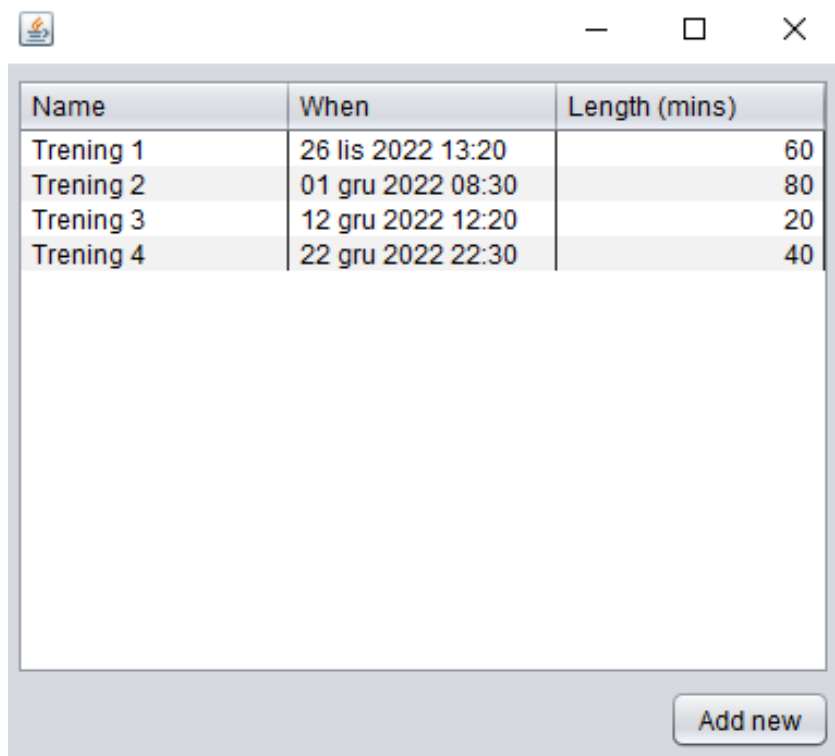
Aplikacja posiada trzy moduły:

1. Moduł główny z tabelą zapisanych treningów (Yuliia Prysiazhna)
2. Moduł dodawania treningu (Darya Benediktovich)
3. Moduł obsługi treningów z zapisywaniem danych do pliku txt (Sebastian Gawron)

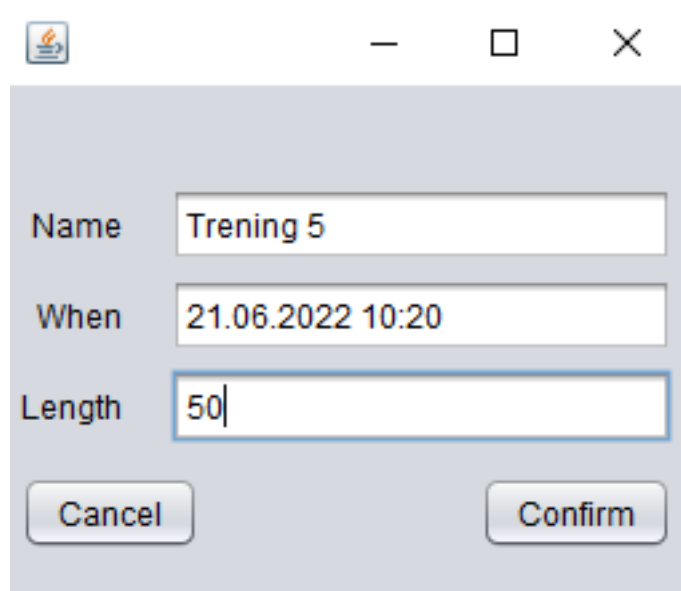
Repozytorium Git: <https://github.com/yuliaprysiashna/group-project>

Wygląd aplikacji:

Okno główne z tabelą treningów, które zostały zapisane do pliku, z możliwością dodania nowego treningu przy naciśnięciu na przycisk „Add new”



Okno dodawania treningu z możliwością zapisu treningu do pliku (przycisk Confirm) lub z możliwością odmiany zapisu (przycisk Cancel)



Name Trening 5

When 21.06.2022 10:20

Length 50

Cancel Confirm

Dane treningów są zapisywane do pliku txt w postaci JSON

```
[{"name":"Test 1","dateTime":"20.12.2021 12:40","minutesLength":40}, {"name":"Test 2","dateTime":"13.02.2022 14:00","minutesLength":25}, {"name":"Test 3","dateTime":"12.01.2022 14:11","minutesLength":90}]
```

Zapis oraz odczyt danych w/z pliku txt

```
public class DefaultTrainingsService implements TrainingsService {  
  
    private final ObjectMapper objectMapper = new ObjectMapper();  
  
    @Override  
    @SneakyThrows  
    public void writeAllTrainings(final List<Training> trainings, final String filePath) {  
        final String json = objectMapper.writeValueAsString(trainings);  
        final Path path = Paths.get(filePath);  
  
        Files.writeString(path, json);  
    }  
  
    @Override  
    @SneakyThrows  
    public ArrayList<Training> readAllTrainings(final String filePath) {  
        final Path path = Paths.get(filePath);  
        final String json = Files.readString(path);  
  
        return objectMapper.readValue(json, new TypeReference<ArrayList<Training>>() {  
        });  
    }  
}
```

Wyświetlenie danych z pliku w postaci tabeli:

```

public class TrainingTableModel extends AbstractTableModel {
    protected static final String[] COLUMN_NAMES = {
        "Name",
        "When",
        "Length (mins)"
    };
    private String filePath = "trainings.txt";
    private TrainingsService trainingsService;
    private ArrayList<Training> rowData;
    public TrainingTableModel() {
        trainingsService = new DefaultTrainingsService();
        File f = new File(filePath);
        if (f.exists() && !f.isDirectory()) {
            rowData = trainingsService.readAllTrainings(filePath);
        } else {
            rowData = new ArrayList<>();
        }
    }

    public void add(Training... pd) { ...
    }

    public void add(ArrayList<Training> pd) {
        rowData.addAll(pd);
        fireTableDataChanged();
        trainingsService.writeAllTrainings(rowData, filePath);
    }
}

```

```

@Override
public Class getColumnClass(int column) {
    return switch (column) {
        case 0 -> String.class;
        case 1 -> LocalDateTime.class;
        case 2 -> Integer.class;
        default -> null;
    };
}

@Override
public Object getValueAt(int i, int j) {
    var t = getTrainingDataAt(i);

    return switch (j) {
        case 0 -> t.getName();
        case 1 -> t.getDateTime();
        case 2 -> t.getMinutesLength();
        default -> null;
    };
}

```